REMARKS

Applicants respectfully request further examination and reconsideration in view of the amended claims and the remarks set forth fully below. In the Office Action dated August 6, 2008, claims 1-49 have been rejected. By the above amendments, the Applicants have amended claims 1, 16, 22, 36 and 39-49, and cancelled claim 38. Accordingly, claims 1-37 and 39-49 are now pending. Favorable reconsideration is respectfully requested in view of the amended claims and the remarks set forth fully below.

Rejections Under 35 U.S.C. §101

Claims 1-8, 16-27 and 36-49 have been rejected under 35 U.S.C. §101 as the claimed invention is directed toward non-statutory subject matter.

It is stated within the Office Action that claims 1-8, 16-27 and 36-38 are directed to a system, but that the body of the claims do not have any hardware support for the claimed systems, within the Office Action it is stated that the Applicant is advised to incorporate "processor" or "computer" to fix the deficiency. By the above amendments, the Applicants have amended the independent claims 1, 16, 22 and 36 to incorporate such elements to fix the deficiency. Accordingly, the Applicant respectfully requests that the Examiner withdraw her rejection under 35 U.S.C. §101.

Claims 2-8, 17-21, 23-27 and 37 are dependent upon the independent claims 1, 16, 22 and 36. Accordingly, the Applicant respectfully requests that the rejections under 35 U.S.C. §101 to these claims also be withdrawn. Claim 38 has been cancelled.

Claims 39-49 are directed to a computer readable signal claim, which is not a physical product. By the above amendments, the Applicants have amended the independent claims 39, 45 and 47 to remove the computer readable signal claim language, and focus on computer readable medium language, which is a physical product, and thereby patentable under 35 U.S.C. §101.

Claims 40-44, 46 and 48-49 are dependent upon the independent claims 39, 45 and 47, respectively. Accordingly, claims 40-44, 46 and 48-49 are allowable as being dependent upon an allowable base claim.

Rejections Under 35 U.S.C. §102

Claims 16-21 have been rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent Publication No. 2002/0093537 to Bocioned (hereinafter Bocioned). The Applicants respectfully disagree with this rejection.

Within the Office Action it is stated that the Bocioned reference teaches a graphical user interface for managing data tasks across multiple applications having a first portion including an enterprise task list, wherein the enterprise task list includes tasks generated outside the systems at the multiple applications and tasks generated inside the system. The Examiner cites Figure 6, paragraphs 25, 7 and 8.

In fact, Bocioned teaches initiating display of a composite window representing a plurality of overlayed tabbed web page (or application) windows [Bocioned, Abstract]. These windows have tabs to identify each application, and not a single window or a "first portion" as described and claimed in the present application [Bocioned, Figure 6, paragraphs 7 and 25]. The system initiates display of a subtask web page (or application window) in the foreground of the composite window in response to user's selection of a visible tab corresponding to the subtask web [Bocioned, paragraph 7, lines 15-20]. Therefore, it is quite obvious that in Bocioned a user selects a tab and a specific application window or web page associated with that tab then is displayed in the graphical user interface while the web pages and/or applications associated with the other tabs are hidden. What is not taught in Bocioned is a single task list in a single viewable portion of a graphical user interface having tasks generated inside and outside the system. In contrast to the teachings of Bocioned, the system and method of the present application includes an enterprise task list in a task manager that is configured to list tasks generated outside of the system through manual entry by a user or system administrator, or third party system, and simultaneously, tasks generated by a task engine within the system.

Claim 16 is directed to a graphical user interface for use in an enterprise information system, wherein the system includes a storage medium, and a processor, and means for displaying the graphical user interface, the graphical user interface for managing data

maintenance tasks across multiple applications, the graphical user interface comprising a first portion including an enterprise task list, said enterprise task list including tasks generated outside the system at the multiple applications and tasks generated within the system. As discussed above, Bocioned does not teach a first portion including an enterprise task list which includes tasks generated outside the system and tasks generated within the system. For at least these reasons the independent claim 16 is allowable over the teachings of Bocioned.

Claims 17-21 are dependent upon the independent claim 16. As discussed above, the independent claim 16 is allowable over the teachings of Bocioned. Accordingly, claims 17-21 are also allowable as being dependent upon an allowable base claim.

Claims 28-29 and 45-46 are rejected under 35 U.S.C. §102(b) as being anticipated by Ada Task Taxonomy Support for Concurrent Programming by Chi et al (hereinafter Chi). The Applicants respectfully disagree with this rejection.

The Chi reference teaches an ada task taxonomy that can be used as a main scheme to help programmers develop concurrent programs. Within the Office Action page 8, section 3.3.3, lines 5-7 are cited to teach populating each of the task data fields that empty with data from corresponding data fields in a task definition template. The Office Action goes on to cite additional portions of section 3.3.3 and section 3.4, claiming that these sections teach the elements of the independent claim 28.

The task taxonomy skeletal ada code, section 3.3.3, teaches that the ada task template is simply used to generate codes for a piece of computer software, not to populate task data fields as described and claimed in the present application. In Chi, the variables in the ada task template can be given, and used to generate ada codes [line 6]. Next, the code generator of the Chi reference generates codes containing specific parameter names when there are any values in the templates which do not have values in the current design. Lastly, it is stated in section 3.3.3 that the template is exactly the description of the super task, i.e., "the templates of all super task levels and the ada codes generated from the task level taxonomy will become a full description of a system design." It is clear from this description that the system described in Chi merely creates a set of computer code from a single ada template that

represents the super tasks, i.e., the entire system design. This section of Chi does not teach populating each of the task data fields that are empty with data from a corresponding data field and a task definition template, and then populating each of the task data fields that are empty with data from corresponding data fields in a task name template.

Next the Office Action cites the reverse engineering tool section 3.4 of the Chi reference to teach populating each of the task data fields that are empty with data from corresponding data fields in a source object template, and populating each of the task data fields that are empty with data from corresponding data fields in a system template. The Office Action cited "points" 3 and 5 of section 3.4. This section of Chi teaches that "the reverse engineering tool offers users the capability to create their own taxonomy from existing ada codes, to create subkind of task from existing definition of taxonomy and existing ada source codes...". "Point" 3 indicates that algorithms can be designed for pattern matching to match token templates generated from taxonomic templates, with token templates generated from ada codes. "Point" 5 teaches interfaces between the results of pattern matching, pattern cluster analysis, and the taxonomic definition editor. It is clear from these teachings that "point" 3 actually teaches taking a code of system design, and reverse engineering to create an actual template. This point does not indeed teach populating each of the task data fields that are empty with data from corresponding data fields in a source object template. The Applicants are confused as to how the Examiner likens reverse engineering code to a template with populating each of the task data fields that are empty with data from corresponding data fields in a source object template. Furthermore, "point" 5 is an interface design tool to create interfaces between the results of pattern matching, pattern cluster analysis, and the taxonomic definition editor. Nowhere can the Applicant find teaching of populating each of the task data fields that are empty with data from corresponding data fields in a system template. Furthermore, in addition to the cited passages of Chi not teaching that which is claimed in the independent claim 28, the Applicants cannot find any teaching of such in the entire Chi reference.

In contrast to the teachings of Chi, the system and method of the present application, referring to Figures 11 and 12, populates a new task data table 210 by first populating each of

the task data fields that are empty with data from corresponding data fields in a task definition template 212. Here, because the task definition template 212 only includes data in the A1 field, the value of that field, "1" is populated in the new task data table 210, field A1. Next, the task data fields that are empty of the new task data table 210 are populated with the task name template 214. Here, because fields A2-A4 of the new task data table 210 are empty, the A1 field of the task name template 214 is not needed, but field A2 of the task name template 214 is populated in the new task data table 210, field A2, with a value of "3". This method is completed with the target object template 216 and the system template 218 as described and claimed in the present application. Once again, the Chi reference does not teach this functionality.

Claim 28 is directed to a method of populating task data fields using task templates, the method comprising the steps of populating each of the task data fields that are empty with data from corresponding data fields in a task definition template; populating each of the task data fields that are empty with data from corresponding data fields in a task name template; populating each of the task data fields that are empty with data from corresponding data fields in a source object template; and populating each of the task data fields that are empty with data from corresponding data fields in a system template. As discussed above, Chi does not teach any of the populating steps of the independent claim 28, accordingly, the independent claim 28 is allowable over the teachings of Chi. The independent claim 45 includes similar limitations to the independent claim 28, and therefore is allowable over the teachings of Chi for the same reasons as discussed above with respect to the independent claim 28.

Claims 29 and 46 are dependent upon the independent claim 28 and 45. As discussed above, the independent claims 28 and 45 are allowable over the teachings of Chi. Accordingly, claims 29 and 46 are also allowable as being dependent upon an allowable base claim.

Claims 1-3, 9, 22 and 36-38 have been rejected under 35 U.S.C. §102(e) as being anticipated by U.S. Patent Publication No. 2002/0188644 to Seidman (hereinafter Seidman). The Applicants respectfully disagree with this rejection.

The Seidman reference teaches a component manager that manages one or more work flow automated task components, wherein the management of these work flow automated task components includes implementing activity coordination with external software subsystems without a developer having to explicitly code to gain such advantages [paragraph 6, lines 15-17, Abstract]. It is clear from the cited passages of Seidman, and a complete reading of the reference, that the Seidman reference is limited to management of a system wherein outside applications create the tasks that are a part of the task lists, or "work list" as described in the reference. The Seidman reference does not teach the task manager having an enterprise task list that includes tasks generated outside the system at multiple applications and tasks generated within the system, and further a task engine module adapted to create tasks, wherein said task engine module is in communication with said task manager module. In other words, Seidman does not teach management of tasks from two different locations, internal and external, nor does it teach creating tasks internally within the system.

In contrast to the teaching of Seidman, the system and method of the present application includes a task manager module that houses an enterprise list that includes tasks from external and internal locations to the system. Furthermore, a task engine module is included that creates tasks and is in communication with the task manager module such that the created tasks may be added to the task enterprise list.

The independent claim 1 is directed to a system comprising an enterprise information system for managing data maintenance tasks across multiple applications within an enterprise, the system comprising a storage medium; a processor; a task manager module configured in the storage medium, and having an enterprise task list, said enterprise task list including tasks generated outside the system at the multiple applications and tasks generated within the system; and a task engine module configured in the storage medium adapted to create tasks, said task engine module in communication with said task manager module, wherein the processor is configured to effectuate the operation of the task manager module and the task engine module. As discussed above, Seidman does not teach a task manager configured to have an enterprise task list including tasks from external and internal locations to the system, and further a task engine module adapted to created tasks. For at least these

reasons, the independent claim 1 is allowable over the teachings of Seidman. The independent claim 36 includes similar limitations to that of the independent claim 1. The Applicant respectfully submits that the independent claim 36 is also allowable over the teachings of Seidman for at least the same reasons as discussed above with the independent claim 1.

Claims 2-3, 9 and 37 are dependent upon the independent claims 1 and 36. As discussed above, the independent claims 1-36 are allowable over the teachings of Seidman. Accordingly, claims 2-3, 9 and 37 are also allowable as being dependent upon an allowable base claim.

The independent claim 22 is directed to a system architecture for centrally managing the creation of tasks comprising a storage medium; a processor; a source object layer configured in the storage medium including one or more source objects, each of said one or more source objects having a corresponding event; a target object layer configured in the storage medium including one or more target objects; and an enterprise task manager layer configured in the storage medium between said source object layer and said target object layer, wherein said enterprise task manager layer centrally manages relationships between each of said one or more source objects and each of said one or more target objects, wherein the processor is configured to effectuate the relationship management. Within the Office Action it is stated that Seidman teaches a system architecture for centrally managing the creation of tasks having a target object layer, an enterprise task manager layer and one or more source object layers. The Examiner cites paragraph 6, lines 22-28 and Figure 7, paragraph 22 however provides very little explanation of these citations. After reviewing the citations, the Applicant is confused how the enterprise work flow automated task being container of Figure 7, and its related description, teaches an enterprise task manager layer. It is further confusing to the Applicant how paragraph 6, lines 22-28 teaches the source object layer including one or more source objects as described and claimed in the present application. For at least these reasons, the independent claim 22 is allowable over the teachings of Seidman. Claim 38 has been cancelled.

Rejections Under 35 U.S.C. §103

Claim 4-8 have been rejected under 35 U.S.C. §103(a) as being unpatentable over Seidman in view of U.S. Patent Publication No. 2003/0045958 to Brandt (hereinafter Brandt). Claims 4-8 are dependent upon the independent claim 1. As discussed above, the independent claim 1is allowable over the teachings of Seidman. Accordingly, claims 4-8 are also allowable as being dependent upon an allowable base claim.

Claims 10-15 and 39-44 have been rejected under 35 U.S.C. §103(a) as being unpatentable over Seidman, in view of U.S. Patent Publication No. 2003/0135384 to Nguyen (hereinafter Nguyen), further in view of U.S. Patent No. 5,625,821 to Record (hereinafter Record), and further in view of U.S. Patent Publication No. 2002/0005894 to Foodman (hereinafter Foodman). The Applicants respectfully disagree with this rejection.

As discussed above, Seidman is a management system for externally generated tasks. Therefore, it does not teach managing internally created tasks. Furthermore, the Examiner characterizes the "work list" in paragraph 19 of Seidman as an event cue, when in fact it is clear from the Examiner's previous citations that the "work list" of Seidman is a task list of externally created tasks [Seidman, paragraph 6].

Furthermore, none of the other cited references in this rejection teach these elements allegedly taught by Seidman. Therefore, the combination does not teach nor make obvious any of these elements.

Claim 10 is directed to a method of generating data maintenance tasks within an enterprise information system, comprising the steps of filing a source object and event to the system; determining whether said event is a custom event or a system event: adding all system events to an event queue; determining for each of said custom events whether a corresponding custom event ruleset is true; adding each of said custom events where said corresponding custom event ruleset is true to said event queue; determining for each of said custom or system events in said event queue whether a task definition corresponding to each of said custom or system events exists; determining for each task definition corresponding to each of said custom or system events whether a task definition ruleset is true; and generating a task for each of said custom or system events having a task definition ruleset that is true.

As discussed above, neither Seidman, Nguyen, Record, Foodman, nor their combination teach the method taught above. For at least these reasons, the independent claim 10 is allowable over the teachings of Seidman. Claim 35 includes similar elements to that of the independent claim 10. Accordingly, claim 39 is also allowable for the same reasons as discussed above with respect to the independent claim 10.

Claims 11-15 and 40-49 are dependent upon the independent claims 10 and 39. As discussed above, the independent claims 10 and 39 are allowable over the teachings of Seidman, Nguyen, Record, Foodman and their combination. Accordingly, claims 11-15 and 40-44 are also allowable as being dependent upon an allowable base claim.

Claim 23 has been rejected under 35 U.S.C. §103(a) as being unpatentable over Seidman in view of Nguyen. Claim 23 is dependent upon the independent claim 22. As discussed above, the independent claim 22 is allowable over the teachings of Seidman. Accordingly, claim 23 is also allowable as being dependent upon an allowable base claim.

Claims 24-27 have been rejected under 35 U.S.C. §103(a) as being unpatentable over Seidman, in view of Nguyen, and further in view of Record. Claims 24-27 are dependent upon independent claim 22. As discussed above, the independent claim 22 is allowable over the teachings of Seidman. Accordingly, claims 24-27 are also allowable as being dependent upon an allowable base claim.

Claims 30-35 and 47-49 have been rejected under 35 U.S.C. §103(a) as being unpatentable over Seidman, in view of ABLE: A Tool Kit for Building Multi Agent Automatic Systems by Bigus et al (hereinafter Bigus), and further in view of Record.

Once again, the Examiner relies upon Seidman to teach automatically generating and performing tasks within a task management system according to a predetermined rules set of an agent. As discussed above, Seidman does not teach this functionality. Furthermore, the Bigus and Record references are not relied upon to teach such functionality, and furthermore they do not indeed teach this functionality. Accordingly, the independent claims 30 and 47 are allowable over the teachings of Seidman, Bigus, Record and their combination.

Claims 31-35 and 48-49 are dependent upon the independent claims 30 and 47. As discussed above, the independent claims 30 and 47 are allowable over the teachings of

Seidman, Bigus, Record and their combination. Accordingly, claims 31-35 and 48-49 are allowable as being dependent upon an allowable base claim.

For these reasons, Applicants respectfully submit that all of the claims are now in a condition for allowance, and allowance at an early date would be appreciated. Should the Examiner have any questions or comments, they are encouraged to call the undersigned at 414-271-7590 to discuss the same so that any outstanding issues can be expeditiously resolved.

Respectfully submitted,

ANDRUS, SCEALES, STARKE & SAWALL, LLP

By _____
Christopher M. Scherer
Reg. No. 50,655

Andrus, Sceales, Starke & Sawall, LLP
100 East Wisconsin Avenue, Suite 1100
Milwaukee, Wisconsin 53202
Telephone: (414) 271-7590
Facsimile: (414) 271-5770